Subject: Re: triangulating over undefined space in irregular grids
Posted by bjelley on Fri, 28 Mar 2008 18:49:46 GMT
View Forum Message <> Reply to Message

On Mar 27, 5:27 pm, bjel...@worldwindsinc.com wrote:
> On Mar 27, 2:56 pm, "ben.bighair" <ben.bigh...@gmail.com> wrote:
>
>
>
>
>
>> On Mar 27, 1:26 pm, bjel...@worldwindsinc.com wrote:
>
>>> On Mar 26, 3:10 pm, "ben.bighair" <ben.bigh...@gmail.com> wrote:
>
>>>> On Mar 26, 3:57 pm, bjel...@worldwindsinc.com wrote:
>
>>>> > On Mar 26, 11:48 am, Kenneth Bowman <k-bow...@tamu.edu> wrote:
>
>>>> > > In article < 419932f8-47d6-4822-aa67-6f6e235ef...@n77g2000hse.googlegroup s.com
>,
>
>>>> > > bjel...@worldwindsinc.com wrote:
>>>> > > > The setup: I am trying to contour plot a number of variables for a
>>>> > > > storm surge grid that has up to 10 km resolution in the open Atlantic,
>>>> > > > yet has 8 meter resolution in New Orleans area waterways. The grid
>>>> > > > obviously includes most anything at or below sea level for the
>>>> > > > Northwest Atlantic domain, but also includes many (200k or so) land-
>>>> > > > based grid points (or it would be useless as a storm surge model). I
>>>> > > > mention the land-based nodes to express that a coastline mask will not
>>>> > > > fix the problem.
>
>>>> > > > The problem: When plotting the results using the triangles returned,
>>>> > > > the result includes vast areas that are outside of the model domain.
>>>> > > > For example, the plot shows data in trangles running from central
>>>> > > > Louisiana to New England, yet the grid does not include any land with
>>>> > > > an elevation greater than 20 meters. So the grid does include some
>>>> > > > land for a range of distances from the coast, from ~5 to ~100 km
>>>> > > > inland based on elevation.
>
>>>> > > I think your main problem is that TRIANGULATE computes the convex hull of
>>>> > > the set of points. That is, there are no "bays" or concave regions in the
>>>> > > resulting set of triangles. This produces the long narrow triangles that
>>>> > > you see across the SE U.S.
>
>>>> > > > Also of importance is that the grid is numbered and ordered in a
>>>> > > > counter-clockwise fashion, not in any order of south to north and the
>>>> > > > like. This is native to the model which uses a finite element method

>>>> > > > for calculations allowing us to successfully resolve conservation of
>>>> > > > momentum, velocity, etc at very high resolutions in areas of interest
>>>> > > > while maintaining the ability to carry out computation at lower
>>>> > > > resolutions over larger areas where mass conservation is
>>>> > > > required...such as the Gulf of Mexico. Could I reorder for purposes of
>>>> > > > plotting? Sure, but do I need to and where would it get me?
>
>>>> > > > I have been through Dr. Bowman's book, Liam Gumley's book, David
>>>> > > > Fanning's site, the astro site, the online help, and the German
>>>> > > > library of IDL routines with no light shed on the solution.
>
>>>> > > > I have fiddled with the tolerance variable passed to triangulate which
>>>> > > > has not changed anything until I make it too large at which time...
>>>> > > > % TRIANGULATE: Points are co-linear, no solution.
>
>>>> > > You might also have some round-off problems, as your grid spacing spans
>>>> > > 4 orders of magnitude 10 to 10^4 m. Are your inputs double precision?
>
>>>> > > Your best bet may be to learn how to use object graphics polygon objects,
>>>> > > but I'm afraid I can't help you there.
>
>>>> > > Ken Bowman- Hide quoted text -
>
>>>> > > - Show quoted text -- Hide quoted text -
>
>>>> > > - Show quoted text -
>
>>>> > Yes, I think that is correct. TRIANGULATE is producing the "convex
>>>> > hull of the set of points"...in your words ;)
>
>>>> > The inputs are all double precision and the data in the files are
>>>> > carried out to enough places to avoid round off errors.
>>>> > Example definition for node 1:"   1 -90.2350725000   30.4780242000
>>>> > -7.9830000000"
>>>> > [node long lat bathymetry]
>>>> > Of course this is far beyond the precision of any measuring system,
>>>> > but that does not matter at the moment.
>
>>>> > The nodes at the corners of the triangles shown in the grid plot
>>>> > (above) are defined in the grid file. I am going to try using that to
>>>> > look for any side of a triangle that is not repeated, which means it
>>>> > is on a boundary, and classify the nodes on either end of the line
>>>> > (side of triangle) as a memebr of an array descibing FAULT_POLYGONS
>>>> > for use in GRIDDATA.
>
>>>> > I'll share how that goes.
>
>>>> Hi,

>
>>>> Wow! What a great problem!
>
>>>> I think it is worth pursuing the FAULT_POLYGONS feature of GRIDDATA.
>>>> I suggest that you run your points through GRID_INPUT first - these
>>>> can be calculated once and then saved for reuse just like the
>>>> triangulation can be saved.
>
>>>> Out of enterprising ignorance I have occasionally introduced sample
>>>> locations over undefined regions like land. To these locations I
>>>> assign whatever the "missing" value is (like NaN). The nice thing
>>>> about that is that the triangulation doesn't have to draw line
>>>> segments that cross undefined regions (like Florida). But I never did
>>>> that for a problem this scale.
>
>>>> It will be very interesting to know how you resolve this - so I am
>>>> glad you will share.
>
>>>> Cheers,
>>>> Ben- Hide quoted text -
>
>>>> - Show quoted text -
>
>>> An update:
>
>>> I am working on the fault_polygons and fault_xy arrays. Some of the
>>> challenge is in connecting the beginning and end of some of the
>>> polygons...the islands are easy. Also working on perfecting the
>>> generation of polygons that do not erroneously connect islands to
>>> mainland. As much as the populace might wish for one, Cuba does not
>>> have a landbridge to Florida ;-)
>
>>> Still perfecting all of that. A little polygon question: I can see
>>> that the fault_polygons can be a concatinated array describing points
>>> in x,y detailed in a 2-by-n array called fault_xy with n = number of
>>> total points in polygons. The question is, can fault_xy be the same
>>> concatinated array, or does it simply need to have all of the points
>>> in the polygons consecutively. In the online help, fault_xy is
>>> described as 2-by-n with n = number of points "that define the
>>> polygon". I am going to have dozens of ploygons and need hundreds more
>>> points in fault_xy than any single polygon contains.
>
>>> I currently have 6000+ polygon points on 61 polygons. A plot and link
>>> forthcoming.
>
>>> I must credit Jim P of ITT VIS for the first suggestion of the
>>> FAULT_POLYGONS in GRIDDATA method via an email response to this
>>> posting.

>
>>> I expect that as soon as I figure this all out David Fanning will come
>>> on and show us an elegant 2 lines to handle it all. I like his
>>> map_gshhs_shoreline routine utilizing some of the logic needed here.
>
>>> Last question for the moment: Is there a way to dictate a polygon of
>>> inclusion (the mainland and open Atlantic boundary) and also numerous
>>> polygons of exclusion (islands)? That would be useful.
>
>> Hi,
>
>> I am not clear on this last question. As I understand things, the
>> interpolation does not occur across a fault. So if you have an
>> "island" there will be interpolation within the island that is
>> independent of the interpolation outside of the island.
>
>> As far as the FAULT_XY thing, I think you simple concatenate the XY
>> coordinates pairs for fault1 to fault2, ...
>
>> Some thing like this...
>
>> ;the polygons
>> xy1 = [[1.0, 1.0], [2.0, 2.0], [2.0, 0.0], [0.0,0.0]]
>> xy2 = [[0.0, 1.0], [3.0, 1.0], [2.5, 4.0], [1.0, 4.0], [0.0, 2.5]]
>> ;paste the together
>> fault_xy = [[xy1],[xy2]]
>> n1 = SIZE(xy1, /N_ELEM)/2
>> n2 = SIZE(xy2, /N_ELEM)/2
>
>> ;start the poly description - terminate with -1
>> fault_poly = [n1, LINDGEN(n1),-1]
>> ;determine the size of the poly descriptor
>> np = SIZE(fault_poly, /N_ELEM)
>> ;add the second polygon descriptor
>> fault_poly = [fault_poly[0:np-2], n2, LINDGEN(n2) + np-2, -1]
>
>> Cheers,
>> Ben- Hide quoted text -
>
>> - Show quoted text -
>
> After running griddata for an hour to generate data for a lame 301x301
> grid, I have decided I am not at all happy with the approach. I have
> since gone the opposite direction with polygons.
>
> I am going to use triangulate for the data to be plotted, as
> originally done, with the expectation that it will not impact my
> contours within the grid domain for real data given the density of

> data points, especially near the land and ocean boundaries. Then I can
> polyfill with the polygons I am generating for the boundaries of the
> grid, including islands, the mainland, and the open ocean boundary. I
> am still working out the addition of a few points in that so the
> polyfill covers the landmass and only the landmass outside of the
> model's grid domain.
>
> I'll show so results soon.
>
> Cheers,
> -Ben- Hide quoted text -
>
> - Show quoted text -

Well here is my latest: http://www.worldwindsinc.com/uploads/wind_polygons.PNG

I have used any unique line between grid points, meaning that it is along a boundary, as part of the polygon and followed along those points to classify a mainland polygon, disallowing the use of any point more than once. I had to force the addition of the mapping extents at the end of the polygon array to give a good fill area in polyfill.

Next, I will do the same for the islands with some minimum distance requirements for the next nearest point to classify as part of a polygon. If I do not go with a minimum distance, multiple islands will appear in a single polygon.

I am confident that the color contour data presented via triangulate does well represent the data within the defined grid area (especially when considering the density of the data points).

This runs in about 90 seconds on my system, a desirable time compared to 1 hour+ for any griddata routine attemped to date. So...problem solved :-)

Cheers and thanks for all you guys could offer on this one,
-Ben